# 生活科におけるプログラミング的思考の育成

作新学院大学人間文化学部特任教授　　高山　裕一

## 1　小学校におけるプログラミング教育の課題

### ⑴今までのプログラミング教育

文科省は，平成３２年度からの新学習指導要領の改訂のポイントの中で，情報活用能力（プログラミング教育を含む）を教育内容の主な改善事項の一つに掲げた。また，総則では，教科等横断的な視点に立った資質・能力の育成の一つの方法として「児童がプログラミングを体験しながら，コンピューターに意図した処理を行わせるために必要な論理的思考力を身に付けるための学習活動」を示し，例示された算数や理科，総合的な学習の時間だけではなく，「例示以外の内容や教科等においても，プログラミングを学習活動として実施することが可能であり，プログラミングに取り組むねらいを踏まえつつ，学校の教育目標や児童の実情等に応じて工夫して取り入れていくことが求められる。」とした。

平成１０年度に公示された学習指導要領では，「コンピューターや情報ネットワークに慣れ親しみ適切に活用する学習活動を充実すること」を，配慮すべき事項として総則で示している。そのため，小学校では特にプログラミング言語としてLogo※１を導入し，「繰り返し」などの命令を使って図形を描写したり，画面上のタートル※２を論理的に動かしたりするなどの学習を行った。しかし，慣れ親しめばよいという目標であったため，プログラミングを行う効果が不明確であったため，当時コンピューターに興味のあった一部の指導者がプログラミングを指導するに留まっていた。

当時，社会の中にパソコンが急速に浸透し，情報ネットワークの速度が増したことで，小学校でもワープロやプレゼンテーション，ネット検索などICTと呼ばれる教育がコンピューター教育と捉えられるようになった。

海外では，イングランドでも日本と同様にコンピューターを道具として教育の中で活用するICT教育が充実していく中でコンピューターサイエンスが深く学習されていないことが指摘され，2014年より「Computing」という教科を新設し，その中で「Computational Thinking」を教科の中で育む思考力とした。

### ⑵これからのプログラミング教育を進めていく上での教育現場の課題

そのような世界の背景や時代の要請を受け，2017年日本でも「小学校段階におけるプログラミング教育の在り方について（議論の取りまとめ）」有識者会議の報告でも「楽しく学んでコンピューターに触れることが好きになることが重要であるが，一方で，楽しいだけで終わっては学校教育としての学習成果に結びついたとは言えず，子供たちの感性や学習意欲に働きかけるためにも不十分である。学習を通じて，子供たちが何に気付き，何を

理解し，何を身に付けるようにするのかといった，指導上のねらいを明確にする必要がある。」とし，小学校におけるプログラミング教育が目指すことの一つとして「プログラミング的思考」を身に付けることとした。それを受け，新学習指導要領ではプログラミング的思考について「自分が意図する一連の活動を実現するために，どのような動きの組合せが必要であり，一つ一つの動きに対応した記号を，どのように組み合わせたらいいのか，記号の組合せをどのように改善していけば，より意図した活動に近づくのか，といったことを論理的に考えていく力」と定義し，子供たちが将来どのような職業に就くとしても時代を超えて求められる力とした。

すなわち，平成３２年度からの小学校現場では，ただ単にロボットやセンサーを使ってものを動かすなどの活動で終わるのではなく，最終的にはプログラミング的思考が身についたのかという観点でプログラミング教育の評価を行うことになる。しかし，小学校現場では，プログラミングを経験している指導者はほとんどいないのに，目的はプログラムを書くことではないと言われても，理解できないのが現状である。また前述の文科省の言うプログラミング的思考の育成についても数時間の総合的な学習の時間や算数，理科の授業でどれだけ身につくのかには大きな疑問が残る。

そこで，全教育活動を通して，プログラミング的思考を育み総合的な学習の時間などにつなげていける①日本の小学校現場の学習内容に即したプログラミング的思考を育むために必要な考える力と②小学校低学年の生活科の活動でのプログラミング的思考を意識させることのできる指導の２点について述べる。

## ２　小学校現場に即したプログラミング的思考を育むための指導

### ⑴イングランドのComputational　Thinking

2014年，イングラドでは今までのICT教育に変わる新しい教科コンピューティングを開始した。教育省の外部機関CAS（Computing At School）では，「Computational　Thinking」について，プログラミング的思考を日本のような長い文章ではなく，６つのコンセプトと５つのアプローチで示している。

＜表１＞　CASの提唱するプログラミング的思考の６つのコンセプトと５つのアプローチ

| | | | | | | |
|---|---|---|---|---|---|---|
| Concept | Logic | Algorithms | Decomposition | Patterns | Abstraction | Evaluation |
| | 手順 | 効率的手順 | 分解 | 型 | 抽象化 | 評価 |
| Approach | Tinkering | Creating | Debugging | Persevering | Collaborating | |
| | 試行錯誤する | 創造する | エラーを訂正する | やり抜く | 協働する | |

⑵プログラミング的思考を育むために必要な考える力

このように，プログラミング的思考をいくつかの具体的な考える力で表すことで，各教科等の多様な学習活動においてこれらの観点を当てはめ意図的にプログラミング的思考を育成することが可能になる。そこで，

① 小学校担任が担当している全ての教科等の内容や小学校現場の授業の方法・活動に即している

② プログラミングを経験していない指導者が理解しやすい言葉で表す

③ 目的が明確で子供にも意識しやすい汎用性のある言葉で活動として表す

以上の３点に留意し，表２のようにプログラミング的思考について４つの考え方とそれぞれに含まれる学習活動（abc）に分類を試みた。

＜表２＞

プログラミング的思考を育むための4つの考える力と学習活動

表中の 手順→単純な動作 仕事→手順が組み合わさった状態 とする

| | |
|---|---|
| **1 構造的に考える力** | |
| 抽象的な問題を分解し具体的にしたり，小さな問題を整理・分類し関係づけたりする | |
| a トップダウン的に分析する(具体化する) | 問題の原因や解決のための方策を広い視野で考え細分化し具体的にする |
| b ボトムアップ的に問題を明確にする(抽象化する) | 複数の事象や言動などから共通点を見いだしたり分類したりする |
| c 関係性を表現する | 事象の関係性や時系列などが一目で理解できるように表す |
| **2 順序性を考える力** | |
| 意図的に手順の順序を考え仕事として分かりやすくする | |
| a 分かりやすい順序にする | 起承転結や，時系列，問題解決的に流れが分かるように仕事を整理する |
| b 無駄のない手順をつくる | 余計な手順や説明を省く |
| a 手順を繰り返したり，分岐させたりする | 手順をより少なく関係性を単純する |
| **3 汎用性を考える力** | |
| 仕事の手順を簡潔にし，応用性のあるものにする | |
| a 仕事を単純化する | 手順を条件と動作に分けて考える |
| b 再現性を意識する | 条件が変わっても，簡単に対応できるように手順を考える |
| c いろいろな場合をイメージする | 手順がどのように使われるか，どのようなミスが起こりえるかを予想する |
| **4 最適化を考える力** | |
| 仕事の手順を評価し，さらに合理化したり新しい手順を生み出したりする | |
| a 新しい手順を創造する(試行錯誤) | いろいろな場合を想定し試しながら，より分かりやすい手順を考え出す |
| a-2 手順を組み合わせる | 使える手順を上手く組み合わせえ，別の手順にする |
| b 合理性を追求する(早く、簡単に，正確に) | 手順をさらに合理的・効率的にしようとする |
| c 客観的に判断する(評価する) | 主観を入れずに，情報から分かることを判断する |

⑶４つの考える力の関係について

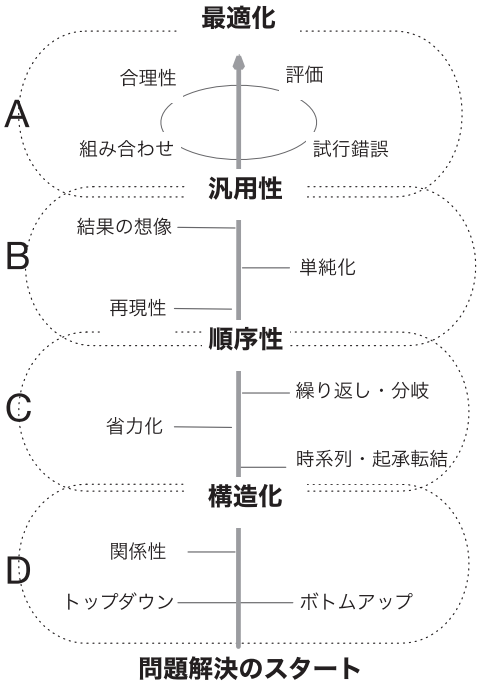
コンピューターのプログラムを作成する場合，プログラムとはコンピューターに対しての命令を順番に与えた命令集であるが，その際作成者が考慮するのは，分かりやすく単純に手順を表すということであり，構造化されたデータの流れと汎用性のあるアルゴリズムの創造が大変重要になってくる。これは，その後のプログラム自身のデバッグや他への活用に対して柔軟に対応できるだけでなく，処理の速さにも関わる。つまり，一度作ったプ

ログラムをいかに最適化できるかと考えることがプログラミング的思考の要と言えよう。

最適化を考えるということは，答えが出ればよいと言うことではなく，図１Aのように常にプログラム自体が効率的，合理的な表現になるように試行錯誤しそれを客観的に評価するPDCAの意識が大切である。

＜図１＞

プログラミング的思考を育む４つの考え方と学習活動の関係




そのためには，プログラム内の一つ一つの手順がデータと切り離されいろいろな場合に対応できるように，単純化と再現性を意識し汎用性のあるものにしていかなければならない。また，どのように使用されその結果どのようなことが起きるかということを広く想像できる力も必要である(図１B)。

このような意識のもと，小さな問題から共通項を見いだしたり問題を具体的に分析したりさらにはそれらの関係性から，問題を構造化して捉え（図１D），一つ一つの手順を分かりやすくまた無駄なく並べて,問題を解決する手順をはっきりさせる(図１C)ことができる力こそがプログラミング的思考である。

このようにプログラミング的思考を育むための力や活動を捉えると，それはまさしく問題解決活動に必要な力であり，文科省からも前述の議論の取りまとめの中で，

> また，「プログラミング的思考には，各教科等で育まれる論理的・創造的な思考力が大きく関係している。各教科等で育む思考力を基盤としながら「プログラミング的思考」が育まれ，「プログラミング的思考」の育成により各教科等における思考の論理性も明確となっていくという関係を考え，アナログ感覚を大事にしていくことの重要性等も踏まえながら，教育課程全体での位置付けを考えていく必要がある。

とあるように，小学校教育の全ての教育活動に当てはめることができるのである。

⑷各教科等におけるプログラミング的思考を育む場と留意点

そこで，問題解決活動とプログラミング的思考を育む４つの考える力を関連させ，各教科等の学習活動の場を当てはめる(表３)と，どの教科等の授業でも行われる単元導入や学習問題作りの場面では，個人の小さな問題を集め学級全体で分類しながら見出しを作る活動や，大きな課題からそれを細分化し一つ一つの学習問題に分析していく活動などは，ものごとを構造的に捉える力が必要であるし，作成した学習問題をどの順に解決していくと

学んだことを生かしていけるか，重複を避け効率よく学習を進めていけるのかといったことは，順序性を考える力を育てるのに好都合な場である。

各教科等でプログラミング的思考を育む場は，指導者が意識することでいくらでも作れるが，教科等によって育てやすさには違いもある。汎用性を考える場は算数科において公式を作る場面や，理科で科学的な知識をもとに物作りを行う場面などでは容易に学習に取り入れることができる。しかし歴史の学習の中で，手順を組み合わせて新たな解決策を創造する場などはイメージしにくいし，無理に取り入れることで教科の目標が達成できない（意識が別に向いてしまう）ことも考えられる。

このように指導者が教科の特性や目標に照らし合わせながら，意図的にプログラミング的思考を使って問題解決を行うことで，効果的に教科本来の授業を深め，その結果プログラミング的思考もさらに深まっていくのである。

だから，例えば総合的な学習の時間でロボット等を用いて問題を探究していく活動を行う場合には，ロボットという新奇性に子供は関心とらわれやすいので，指導者がその探究活動をとおしてどのようなプログラミング的思考を育もうとしているのかをしっかり捉えておかないと総合としてもプログラミングとしてもどちらの目標も達成できないことになってしまうであろう。

また，総合的な学習や特別の教科道徳，さらには図工や音楽などの芸術的な教科の内容によっては，子供の主観を大切にする活動やオープンエンドの活動など，プログラミング的思考が苦手としている場もあることも指導者は十分に留意する必要がある。

＜表３＞

小学校各教科等の中で プログラミング的思考を育てられる場の例

| プログラミング的思考に必要な力 | 問題解決活動の場 | 小学校の各教科等（除 生活科） | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 国語 | 社会 | 算数 | 理科 | 音楽 | 図工 | 家庭 | 体育 | 総合 | 特活 | 道徳 |
| 1）構造的に考える力 | | | | | | | | | | | | |
| a トップダウン的に分析する(具体化する) | 単元導入 | | ◎歴史・社会問題 | | | | ○イメージを深める | ○問題作り | | ◎問題作り | ◎学級の問題解決 | ◎気持ちの分析 |
| b ボトムアップ的に問題を明確にする（抽象化する） | 体験活動 | ◎物語の感想 | | ◎公式 | ◎観察から分類 | ○音からの曲の創造 | | | ◎ゲームの反省 | ◎問題作り | ◎議題作り | ◎問題の意識化 |
| c 関係性を表現する | 話合い | ◎説明文 | ◎歴史・社会問題 | ◎比例，反比例 | ◎実験（予想～考察） | | | | ○ | ◎話合いの整理 | | ◎問題把握 |
| 2）順序性を考える力 | | | | | | | | | | | | |
| a 分かりやすい順序にする | 単元の見通し | ◎長文の把握 | ◎歴史・社会問題 | ◎証明 | ◎実験 | ◎曲の展開 | ○製作 | | | ◎見学 | | |
| b 無駄のない手順をつくる | 活動後の整理 | | ○社会科見学 | ◎求積 | ○観察 | | | ◎調理 | | | | |
| a 手順を繰り返したり，分岐させたりする | 思考中考察 | ○ | ◎人々の工夫 | | ◎知識の利用 | ○曲の構成 | | ◎家の仕事 | | ◎役割分担 | ◎役割分担 | |
| 3）汎用性を考える力 | | | | | | | | | | | | |
| a 仕事を単純化する | 問題発見 | ○文章の解釈 | | | ○予想 | | | ○ | ◎チームの役割 | | ◎係活動 | |
| b 再現性を意識する | 結論 深化 | | | ◎公式化 | ◎科学的 | ◎和音 | | ◎実践力 | | | ◎委員会活動 | |
| c いろいろな場合をイメージする | 予想 考察 | | ○人々への対応 | | ◎知識の利用 | | | ◎家族 | ◎作戦 | ◎考察 | | ◎他への心遣い |
| 4）最適化を考える力 | | | | | | | | | | | | |
| a 新しい手順を創造する(試行錯誤) | 解決策の検討 | | ○人々への対応 | ○解決法 | | ◎作曲 | | | ◎ゲーム | | ○解決法 | |
| a-2 手順を組み合わせる | 深化 発展 | | | ◎図形の面積 | ○ | ○ | ○ | ◎掃除，調理 | | | ◎学校行事 | ◎自分の出来ること |
| b 合理性を追求する(早く、簡単に，正確に | 自己解決後 | | | ◎数学的な追究 | ○実験方法 | | | | | | | |
| c 客観的に判断する(評価する) | 思考や活動後の評価 | ◎結論 | ◎結論 | ◎数学的な追究 | ◎結論 | | | | ◎ゲームの反省 | | | ◎自分の意識の変 |

## 3　生活科におけるプログラミン的思考の育成

### ⑴　新学習指導要領における生活科での思考

平成２８年８月に中央教育審議会初等中等教育分科会教育課程部会から出された「次期学習指導要領等に向けたこれまでの審議のまとめ」では，生活科の見方・考え方と育成する資質・能力について以下のように述べられている。

ｉ)教育課程の示し方の改善

ア 資質・能力を育成する学びの過程についての考え方

（１番目の○は略）

○ 具体的な活動や体験を通して，比較したり，分類したり，関連付けたりなどして解釈し把握するとともに，試行したり，予測したり，工夫したりなどして新たな活動や行動を創り出すことを通して，自分自身や自分の生活について考え，個別的な気付きが関係的な気付きへと質的に高まるなど，新たな気付きを生み出すことが期待される。（アンダーライン筆者）

ここで述べられている「比較，分類，関連づけ」はまさに構造化にともなう活動であり，「試行，予測，創意工夫」は，最適化に向けての活動である。

生活科はもともと子供の問題解決活動から自己の気付きの質を高めていく教科であるので，他教科と比べてもプログラミング的思考とは関連の深い教科である。

また，物事を構造化・順序化し，学んだことを他へ利用できる汎用性を求めていく態度は，新学習指導要領で言う「学びにむかう力」の大切な要素となるものである。

### ⑵　生活科におけるプログラミング的思考を意識できるようにするための支援

前述のように生活科の諸活動はプログラミング的思考と深い関係にあるので，生活科の活動を行っていれば後は指導者がプログラミング的思考の理解と育てる意識があれば，それほど難しいことではない。

ただ，生活科は小学生としては幼い低学年時に行う教科であるので，概念形成の乏しい子供にとってトップダウンの考え方は理解しにくいことがあるなど，発達段階を十分に考慮することや，他の思考力と同じように繰り返し継続的に指導することで身につけるものであるという捉え方が必要である。

そこで生活科の活動の中で子供がプログラミング的思考を意識できるような支援の一つとして，汎用性のある分かりやすいキーワードを入れた助言を作成した。（表４）

低学年の子供にとって，例えば「似ているところを見つけて仲間分けをしよう」と投げかけてもその活動が抽象化につながっているという意識はない。しかし，仲間分けという活動が問題を明確にするときに役立つという経験を繰り返すことによって，中・高学年生になって複雑な問題も構造化できるという力となっていく。

このように，プログラミング的思考の考え方は，子供自身が自ら問題解決を行っていくときの基本的な力になることを意識して，生活科の各活動で意図的に投げかけられるよ

うにしていきたい。

＜表４＞

| 生活科の活動中に プログラミング的思考を意識出来るようにするための助言のキーワード | | |
|---|---|---|
| プログラミング的思考を育む4つの考える力と活動 | | 生活科の学習の中で，考えるきっかけづくりを作るための キーワード（投げかけの言葉） |
| 1) 構造的に考える力 | | |
| | a トップダウン的に分析する（具体化する） | 自分がやりたいことを**はっきり** させよう |
| | b ボトムアップ的に問題を明確にする（抽象化する） | **にてるところを見つけて**仲間分けしよう |
| | c 関係性を表現する | **一目で分かる**ように表そう |
| 2) 順序性を考える力 | | |
| | a 分かりやすい順序にする | **順序よく**並べて整理しよう |
| | b 無駄のない手順をつくる | よけいなことを省いて**すっきり** させよう |
| | a 手順を繰り返したり，分岐させたりする | **簡単にできる**マニュアルを作ろう |
| 3) 汎用性を考える力 | | |
| | a 仕事を単純化する | **できるだけ簡単**にまとめよう |
| | b 再現性を意識する | 次に生かせる**ツールを**作ろう |
| | c いろいろな場合をイメージする | **相手を考えて** どうなるか予想しよう |
| 4) 最適化を考える力 | | |
| | a 新しい手順を創造する（試行錯誤） | **反省を生かして** 新しい方法を考えよう |
| | a-2 手順を組み合わせる | **情報を取り入れ**て，もっとよくしよう |
| | b 合理性を追求する（早く、簡単に，正確に） | **もっと早く・簡単に** を考えよう |
| | c 客観的に判断する（評価する） | 結果について，**はなれたところから**見てみよう |

⑶生活科の主な活動におけるプログラミング的思考を育める場

そこで，生活科の各活動と今まで述べたプログラミング的思考を育むための４つの考える力，助言の中のキーワードを問題解決活動の流れに従って整理したものが表５である。

ここでは，生活科の各内容の中で特に時間をかけて行われる

① 教室から出て調べる活動

② 自然や動植物と触れ合う活動

③ オモチャなどを作って遊ぶ活動

④ いろいろな人と関わる活動

の４つに絞って整理した。（表５）

表５からも分かるようにプログラミング的思考を意識できるようにするためという見方からの学習活動や留意する場や助言などの支援は，とりもなおさず生活科の活動そのものを深め子供の質の高い気付きを促す支援にもなっていくことに気付く。

＜表５　生活科の主な活動におけるプログラミング的思考を育める場＞

| 問題解決活動の流れ | | プログラミング的思考を育む４つの考える力と学習活動 | | 発達段階としての難易度 | 生活科の学習の中で，考えるきっかけを作るためのキーワード（投げかけの言葉） |
|---|---|---|---|---|---|
| 単元導入 | 導入時，何が問題なのかを気付いていく場面 | 構造化 a トップダウン的に分析する(具体化する) | 問題の原因や解決のための方策を広い視野で考え細分化し，より具体的にする | ○ | 自分がやりたいことを<br>はっきり　させよう |
| 体験活動 | 自分自身が解決する課題と問題の関係を捉える場面 | 構造化 b ボトムアップ的に問題を明確にする（抽象化する） | 複数の事象や言動などから共通点を見いだしたり分類したりする | ◎ | にてるところを見つけて<br>仲間分けしよう |
| 見通し<br>（活動後の整理 | 何をどのような順で解決するか計画する場面 | 順序性 a 分かりやすい順序にする | 起承転結や，時系列，問題解決的に流れが分かるように仕事を整理する | ◎ | 順序よく並べて整理しよう |
| 予想<br>考察 | 解決するための方法やその結果を予想する場面 | 汎用性 c いろいろな場合をイメージする | 手順がどのように使われるか，またどのようなミスが起こりえるかを予想する | ○ | 相手を考えて　どうなるか予想しよう |
| 試行錯誤<br>（考察） | 解決活動中，結果を評価しまた活動する場面 | 順序性 a 手順を繰り返したり，分岐させたりする | 手順をより少なく関係性を単純する | ○ | 簡単にできるマニュアルを作ろう |
| 話合い<br>(収集・整理) | 解決した結果を収集し，整理しながら他との関係性をみつける場面 | 構造化 c 関係性を表現する | 事象の関係性や時系列などが一目で理解できるように表す | ○ | 一目で分かるように表そう |
| 話合い<br>(精選・結論) | | 順序性 b 無駄のない手順をつくる | 余計な手順や説明を省く | ▲ | よけいなことを省いて<br>すっきり　させよう |
| 自己解決後 | 解決した自分なりの答えについて，さらによくならないかを検討する場面 | 最適化 b 合理性を追求する（早く、簡単に，正確に） | 手順をさらに合理的・効率的にしようとする | ▲ | もっと早く・簡単に　を考えよう |
| 思考中<br>解決後の評価 | 自分の答えが満足できるものかを最初の予想段階や，他の答えと比べて評価する場面 | 最適化 c 客観的に判断する（評価する） | 主観を入れずに，情報から分かることを判断する | ▲ | 結果について，はなれたところから見てみよう |
| 解決策の検討 | 問題解決の結論から新たな課題を発見し，解決方法を模索する場面 | 最適化 a 新しい手順を創造する（試行錯誤） | いろいろな場合を想定し試しながら，より分かりやすい手順を考え出す | ◎ | 反省を生かして　新しい方法を考えよう |
| 深化<br>発展 | | 最適化 a-2 手順を組み合わせる | 使える手順を上手く組み合わせえ，別の手順にする | ◎ | 情報を取り入れて，もっとよくしよう |
| 問題発見 | 何度かの問題解決を繰り返した結果から見えてきた解決方法から，誰にでも使える方法を考える場面 | 汎用性 a 仕事を単純化する | 手順を条件と動作に分けて考える | ▲ | できるだけ簡単にまとめよう |
| 結論<br>深化 | | 汎用性 b 再現性を意識する | 条件が変わっても，簡単に対応できるように手順を考える | ○ | 次に生かせるツールを作ろう |

| 生活科の主な活動の中での，プログラミング的思考を育てられる場の例 | | | |
|---|---|---|---|
| 探検する | 栽培・飼育・自然 | 遊ぶ | 人とふれあう |
| 学校（内・外）<br>町（通学路・公園・店）<br>乗り物に乗って（公共施設） | 花　野菜<br>昆虫　魚　ウサギ<br>四季の変化 | 遊具<br>おもちゃづくり<br>昔のあそび | 学校内（友達・異学年・先生）　家庭<br>学校外（地域・お年寄り） |
| ・探検で自分の見てみたいことややってみたいことを，はっきりさせる | ・季節などの条件から，自分が何を育てたり見つけたりすればよいのかをはっきりさせる | ・誰と，どこで遊ぶかなどの条件を意識して，遊びの種類や遊び方を考える | ・人と触れ合うときに起きる問題について，相手の年齢やどこの人か等によって違うことを見つける |
| ・いろいろな教室での発見を整理すると共通点が見つける | ・みんなが見つけてきたもの集めてきたものから，場所や季節ならではの特徴に気付く | ・遊びをする場所などを遊び方を考えながら分けていく | ・人に接するときのマナーについて，いろいろな問題から気をつけることを分類する |
| ・，行きたいところ，時間，仲間などの条件によって探検の経路を変える | ・植物の成長の様子や，四季の自然の移り変わりなどを，時系列を意識して表す | ・遊び方を教えるときの説明を，準備や練習の仕方などを考えながら，時系列に表す | ・人と話すときに自分の伝えたいことを順序よく話せるようにメモを作る |
| ・探検時に出会う人や公共の乗り物などについて，危険などを予め予想して，対応を考える | ・野菜の栽培や昆虫や動物の飼育でのえさや水の量などを相手の状況を考えながら調節する【相手は？】 | ・作ったオモチャがどんな箇所で上手くいかないかや安全性などを考えて，作り方を工夫する | ・基本的な挨拶や言葉遣いなどをどのような場合に使うか考える |
| ・効率よく見学できるように，歩く順序を工夫する<br>・公共の乗り物に乗る手順について，確認する。 | ・生き物の世話の仕方について，順序や状況に応じて変えることなどをマニュアル化する | ・遊びの大会の企画がスムーズに行くためにプログラムを調整する。 | |
| ・探検してきた場所を地図上に表すことで，一目で自分の活動を振り返れること | ・野菜の栽培や，四季の変化などを時系列で表す<br>・自然の花などの見つけた場所を地図に表す | ・遊びの大会会場の配置などを地図に表す | |
| ・地図上に表した探検の経路から，探検に関係ないことを抜くこと | ・生き物の世話について，必ずしなくてはならないことと，そうでないことに分ける | 遊びの仕方をお客に分かりやすく伝えるために，内容を精選する | ・人と接するときの留意点を整理してまとめる。 |
| ・探検するときに準備する物や，注意することについて，最低限必要なことをまとめる。 | ・生き物の世話について，もっと簡単にできる方法を工夫する | ・大会の景品作りなどたくさんのものを作るときに，早く簡単にできる方法を考える | |
| ・出かける前に予想していた探検の経路や，活動と結果を比べる | ・野菜の収穫量について，世話の仕方などを振り返って考察する | ・自分で楽しむだけでなく，相手にとってどうだったかと言う見方で，自分で作った遊びを見直す | |
| ・他の人の結果も参考にして探検の反省をもとに，2回目以降の探検について，活用内容や経路を工夫する | ・反省を上手く生かして，2回目以降の野菜や花の栽培，世話を工夫する | ・相手のことや安全性，おもしろさなどを考え，おもちゃを改善する | ・いろいろな相手との触れ合いから，年齢やどんな人なのかを考えて，対応を判断する |
| ↓ | ・2回目以降の野菜や花の栽培，世話を工夫について他の人の結果や資料を参考にする | ・いろいろな参考書や友達のオモチャなどから，自分のオモチャに取り入れている | |
| ・次に便利なように，いろいろな探検時に気をつけることを一覧にしておく | ・四季のそれぞれの自然のようすについて簡単な説明を考える | ・遊び方やルールをだれにでも分かりやすくなるように簡単にしている | |
| ・学校探検をもとに，公園や公共施設で生かせるマナーなどを見つける | ・次に役立つ野菜や花の栽培マニュアルを作る | ゲームで高得点が採れる方法を考えたり，作ったオモチャを別な遊びに生かしている | |

## 5　最後に

今回の新学習指導要領で言うプログラミング教育について，現場では高学年の算数や理科，総合的な学習の時間の一部の単元で行えばよいと言う誤った理解や，ロボットに命令を与え動かせばよいという学びのない這い回る活動が広がっていく懸念がある。

２０年前，Logo を導入したときと同じことにならないように，まず「プログラミング教育はその基盤となるプログラミング的思考の育成を目的としている」ということを，低学年から全教育活動を通して指導者が意識できる研修を速やかに開始することが望まれる。

※ １　Logo

1967 年に米マサチューセッツ工科大学の Seymour A.Papert 氏によって開発された，特に算数の図形の概念を形成するのに有効なプログラミング言語。

命令は，前後と進行方向（角度）が基本で，それをもとに手順を作成していくことで自然に構造的プログラミングを作ることが出来た。

また，エラーメッセージが「その手順は知らない」という表現で出されるため，子供はより具体的に物事を分析することを求められ，トップダウン的な思考を身につけることができた。

※ ２　タートル

Logo では，画面上の亀（タートル）がカーソルとなっていろいろな図形を描くようになっていた。描かれた図形は見方を変えれば，自分のプログラミングの全ての結果を返すものでありすぐに評価・再検討という試行錯誤が自然にできた。

現在，プログラミング教育ではロボットを使っていろいろな仕事をさせるが，途中経過が表せないため子供は結果のみで評価をしてしまい，最適化を求めるプログラム自体の評価が疎かになることが懸念される。

## 参考文献

１　新学習指導要領　2017　文部科学省

２　Computing Progression Pathways　2015　CAS(Computimg At School)

３　小学校段階におけるプログラミング教育の在り方について（議論の取りまとめ）
2017　文部科学省　有識者会議